DOCUMENT RESUME

ED 326 582 TM 015 954

AUTHOR Adema, Jos J.
TITLE A Revised Simplex Method for Test Construction Problems. Research Report 90-5.
INSTITUTION Twente Univ., Enschede (Netherlands). Dept. of Education.
PUB DATE Sep 90
NOTE 44p.
AVAILABLE FROM Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.
PUB TYPE Reports - Evaluative/Feasibility (142)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Computer Assisted Testing; Equations (Mathematics); Foreign Countries; *Item Banks; Item Response Theory; Linear Programing; *Mathematical Models; *Test Construction
IDENTIFIERS *0 1 Linear Programming Model; LINPROG Computer Program; *Simplex Models

ABSTRACT

Linear programming models with 0-1 variables are useful for the construction of tests from an item bank. Most solution strategies for these models start with solving the relaxed 0-1 linear programming model, allowing the 0-1 variables to take on values between 0 and 1. Then, a 0-1 solution is found by just rounding, optimal rounding, or a heuristic. In most applications, the latter can be executed very rapidly. This paper uses the revised simplex method to solve the relaxed 0-1 linear programming method for test construction. The simplex method is modified such that the characteristics of test construction problems are taken into account. The modifications were implemented in the computer program LINPROG. Two item banks, each containing 450 items, were generated to determine if central processing unit (CPU) time was gained. Computational experiments showed a gain of CPU time for most modifications. Ten tables present the results for the modifications. (Author/SLD)

A Revised Simplex Method for Test Construction Problems

Research Report 90-5

Jos J. Adema

Department of EDUCATION

Division of Educational Measurement and Data Analysis

University of Twente

Colofon:
Typing: L.A.M. Bosch-Padberg
Cover design: Audiovisuele Sectie TOLAB
Toegepaste Onderwijskunde
Printed by: Centrale Reproductie-afdeling

A Revised Simplex Method

for Test Construction Problems

Jos J. Adema

A revised simplex method for test construction problems , Jos J. Adema - Enschede : University of Twente, Depart ment of Education, September, 1990. - 36 pages

Abstract

Linear programming models with 0-1 variables are useful for the construction of tests from an item bank. Most solution strategies for these models start with solving the relaxed 0-1 linear programming model, that is, the 0-1 variables are also allowed to take on values between 0 and 1. Then, a 0-1 solution is found by just rounding, optimal rounding, or a heuristic. In most applications the latter can be executed very fast. This paper uses the revised simplex method to solve the relaxed 0-1 linear programming model for test construction. The simplex method is modified such that the characteristics of test construction problems are taken into account. The modifications were implemented in the computer program LINPROG. Computational experiments showed a gain of CPU time for most modifications.

Keywords: Test Construction, Item Banking, Linear Programming, Revised Simplex Method

A Revised Simplex Method
for Test Construction Problems

Developments in item response theory and computer science have made it more convenient to build item banks. An interesting application of item banking is the construction of customized tests, that is, the selection of a test to meet the consumer's demands. Theunissen (1985) has shown that the problem of selecting items for a test (the test construction problem) can be solved using 0-1 linear programming. Generally, in linear programming a problem is translated into a model, which consists of a linear objective function and a number of linear constraints. For instance, in the 0-1 linear programming model by Theunissen, the objective was minimization of the number of items in the test under the constraints that the amount of test information at a few specified ability levels be larger than a prespecified quantity. Until now, most research has been directed to the problem of modeling a test construction problem as a 0-1 linear programming model, and the problem of solving such a model has been given less attention in the literature on test theory.

A linear programming problem without integer variables is solvable by the well-known simplex method. The simplex method was invented by G.B. Dantzig. A report of the development of the simplex method is given in Dantzig (1963). In computer codes, mostly the revised simplex method is used.

A description of the revised simplex method is given in this paper. The 0-1 linear programming problem is a special form of the integer linear programming problem, which can be solved by applying a branch-and-bound method (Land & Doig, 1960). In a branch-and-bound method the simplex method is used repeatedly. Therefore, the method is time consuming. To avoid this problem, other solution strategies for solving test construction problems have been proposed such as rounding, optimal rounding (van der Linden & Boekkooi-Timminga, 1989) and a heuristic (Adema, Boekkooi-Timminga & van der Linden, in press). In these solution strategies the relaxed 0-1 linear programming problem; that is, the problem in which the variables are allowed to take on values between 0 and 1, is solved first. Then, given the solution to the relaxed problem, a good suboptimal 0-1 solution is computed very fast.

If one wants to construct tests in an interactive mode, a fast method for solving the relaxed problem is needed, because this makes it more convenient for the test constructor to specify his/her demands, view the test, and possibly adjust the demands in one session. It is, therefore, interesting to study the possibility of reducing waiting times by implementing the simplex method for test construction such that CPU time reduces and minimal computer storage capacity is needed. The latter is important if one wants to contruct tests on a personal computer. In this paper implementations of the revised simplex method are presented

taking the special form of the 0-1 linear programming models for test construction into account. In particular, the Maximin Model (van der Linden & Boekkooi-Timminga, 1989) will be regarded, because this model has the advantage that the test constructor does not have to specify an absolute target test information function but only the relative shape of it.

In the next section the Maximin Model is given. This section is followed by a description of the revised simplex method. Then, some modifications for the revised simplex method are given. Finally, the practical gain of the implementations in CPU time is shown.

Maximin Model

In this section the Maximin Model is formulated. Define the decision variables x_i as

$$x_i = \begin{cases} 0 \text{ item } i \text{ not in the test} \\ 1 \text{ item } i \text{ in the test,} \end{cases} \qquad i = 1, \ldots, I,$$

where I is the number of items in the item bank. Let $I_i(\theta_k)$, $k = 1, \ldots, K$; $i = 1, \ldots, I$ be the information of item i at ability level θ_k. The proportion of information required at ability level θ_k is specified by r_k. The vector $\{r_k\}$ constitutes a target for the test information function; the latter is considered discrete here to make it possible to

formulate the problem as a 0-1 linear programming problem. The decision variable y determines the vertical location of the test information function. If N is the number of items to be selected for the test, then the Maximin Model can be written as follows (the presentation of the model is followed by an explanation):

(1) maximize y,

subject to

(2) $\sum_{i=1}^{I} I_i(\theta_k)x_i - r_k y \geq 0,$ $k = 1, 2, \ldots, K,$

(3) $\sum_{i=1}^{I} x_i = N,$

(4) $\sum_{i=1}^{I} v_{ij}x_i = w_j,$ $j = 1, 2, \ldots, J,$

(5) $x_i \in \{0,1\},$ $i = 1, 2, \ldots, I,$

(6) $y \geq 0.$

In the objective function (1) the vertical location of the test information function is maximized. By the constraints in (2) $(r_1 y, \ldots, r_K y)$ is a series of lower bounds to the test

information function $I(\theta_k)$ at ability levels θ_k. The constraints in (4) are added as a general provision to deal with practical constraints, for instance, on test composition, administration time, and the like; for examples, see Adema and van der Linden (1989). Each different application of (4) will involve different definitions of v_{ij} and w_j.

The Revised Simplex Method

In this section the revised simplex method as described in Murtagh (1981) is briefly reviewed. The revised simplex method is the standard for computer codes for solving linear programming problems, and the method is introduced in most text books (e.g., Papadimitriou & Steiglitz, 1982). Generally, a linear programming problem with n variables and m constraints can be written as:

$$(7) \qquad \text{maximize } \mathbf{c}^T\mathbf{x} \ (= \sum_{i=1}^{n} c_i x_i),$$

subject to

$$(8) \qquad \mathbf{A}\mathbf{x} = \mathbf{b} \qquad (\sum_{i=1}^{n} a_{ji} x_i = b_j, \qquad j = 1, 2, \ldots, m),$$

(9) $\mathbf{x} \geq 0 \qquad (x_i \geq 0, \qquad i = 1, 2, \ldots, n),$

where (7) is the objective function and equations (8) and (9) are constraints. Notice that an inequality constraint

$$\sum_{i=1}^{n} a_{ji}x_i \leq b_j,$$

can be written as an equality constraint by introducing a nonnegative variable as follows:

$$\sum_{i=1}^{n} a_{ji}x_i + s_j = b_j,$$

where $s_j \geq 0$. The variable s_j is called a slack variable.

If it is assumed that $m < n$, matrix A can be partitioned into submatrices B and D

$$A = [B|D],$$

where B is a $m \times m$ nonsingular submatrix. Given this partition equations (8) can be written as

$$B\mathbf{x}_B + D\mathbf{x}_D = b,$$

where $\mathbf{x}$ has been partitioned into

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_B \\ \mathbf{x}_D \end{bmatrix}$$

corresponding to the partition of A. Similarly, $\mathbf{c}$ is partitioned into

$$\mathbf{c} = \begin{bmatrix} \mathbf{c}_B \\ \mathbf{c}_D \end{bmatrix}.$$

Since B is nonsingular $\mathbf{x}_B$ can be written as

$$\mathbf{x}_B = B^{-1}\mathbf{b} - B^{-1}D\mathbf{x}_D.$$

The m variables in $\mathbf{x}_B$ are called the basic variables, because they are solved in terms of the other n-m variables (nonbasic variables) in $\mathbf{x}_D$. A basic solution is defined as one in which the nonbasic variables are set at their bounds, in this case to zero. A basic feasible solution is a basic solution in which all the terms of the vector $B^{-1}\mathbf{b}$ are nonnegative. Remark, that a large number of partitions for A into B and D exists. The general idea of the simplex method is to search through the basic feasible solutions by moving from one basic feasible solution to an adjacent one with a better objective function value. In each iteration a basic variable leaves the basis and a nonbasic variable enters the

basis, that is, new matrices B and D are chosen. This process continues until no further improvement can be obtained.

Now the steps of the revised simplex method are given. For the logic underlying these steps and a more detailed description of the steps the reader is referred to the operations research literature (e.g., Murtagh, 1981).

Step 1: Produce a pricing vector. Evaluate $\pi^T = \mathbf{c}_B{}^T B^{-1}$, where π^T is called the pricing vector.

Step 2: Price out the nonbasic variables and select the entering nonbasic variable. In the standard form of the revised simplex method $d_i = \pi^T \mathbf{a}_i - c_i$ is evaluated for each nonbasic variable. It is commom to refer to d_i as the reduced cost of variable i. The variable with the most negative reduced cost d_i is chosen as the nonbasic variable entering the basis. If no reduced cost is less than zero, then STOP (an optimal feasible solution is found).

Step 3: Find the leaving basic variable. The computations necessary to select the basic variable that leaves the basis are executed.

Step 4: Pivot. In this step the new B^{-1} is computed. Go to Step 1.

A detailed description of Steps 3 and 4 is not given here, because modifications with respect to these steps are not made in this paper.

Modifications in the Revised Simplex Method

The fully relaxed version of a 0-1 linear programming problem is the problem with all constraints $0 \leq x_i \leq 1$ instead of $x_i \in \{0,1\}$. If for a test construction problem the related fully relaxed problem is solved, a good suboptimal 0-1 solution can be found very fast by methods as rounding and optimal rounding. The fully relaxed problem is solvable by the simplex method. Thus, to solve a test construction problem quickly it is important to have a good implementation of the simplex method. It is the purpose of this paper to present implementations of the simplex method that speed up the calculations of the solutions considerably and save memory space.

Pricing Strategies

In Step 2 of the revised simplex method an entering nonbasic variable has to be chosen. There are several possibilities of choosing this variable; a few of them will be considered in this paper. Generally, the success of pricing strategies depends on the linear programming problems considered.

Strategy 1: The standard strategy. The variable with the most negative reduced cost is chosen.

Strategy 2: Starting from the last selected variable the first variable with negative reduced cost is selected (see, e.g., Syslo, Deo & Kowalik, 1983, p.14).

Strategy 3: The P variables with the most negative reduced costs are selected. Then Strategy 1 is applied to these P variables until all P variables have nonnegative reduced cost. Again, the P variables with the most negative reduced costs are selected. Etcetra (see, e.g., Lasdon, p.311).

Strategy 4: Partial pricing. In this strategy only a part of the variables is considered, namely the next P variables after the last variable for which the reduced cost was computed in the previous iteration (see, e.g., Hartley, 1985, p.64).

Strategy 1 is mostly used in the revised simplex method. The other pricing strategies might be better with respect to CPU time, because the computational burden in Step 2 is reduced especially for problems with many variables. On the other hand, the number of iterations will probably increase so that it is not sure that Strategies 2 through 4 will perform

better. Strategies 1 and 2 are special cases of Strategy 4, because Strategy 1 is Strategy 4 with P equal to the number of variables (including slack variables) and Strategy 2 is Strategy 1 with P = 1. Strategy 1 is also equal to Strategy 3 with P = 1.

Practical Constraints

Van der Linden and Boekkooi-Timminga (1989) have given an overview of possible practical constraints, for instance, constraints on administration time and composition of the test. Some of the constraints consider items which belong to the same subdomain of the item bank. This implies that the columns in matrix A corresponding to these items are partly identical.

Example:

Suppose we have an item bank for French with 450 items. The item bank is divided in three subdomains with respect to its content:

Items 1-150: vocabulary items;

Items 151-300: grammar items;

Items 301-450: reading comprehension items.

The first 80 items of each subdomain are assumed to be of the multiple choice type; the other items are matching items.

Now suppose a test constructor wants to have a test with the following composition:

1) The test should contain 10 vocabulary, 10 grammar, and 10 reading comprehension items.
2) Exactly 15 multiple choice and 15 matching items should be included in the test.

The following constraints represent this composition:

$$(10) \qquad \sum_{i=1}^{150} x_i = 10,$$

$$(11) \qquad \sum_{i=151}^{300} x_i = 10,$$

$$(12) \qquad \sum_{i=301}^{450} x_i = 10,$$

$$(13) \qquad \sum_{i=1}^{80} x_i + \sum_{i=151}^{230} x_i + \sum_{i=301}^{380} x_i = 15,$$

$$(14) \qquad \sum_{i=81}^{150} x_i + \sum_{i=231}^{300} x_i + \sum_{i=381}^{450} x_i = 15.$$

If we add these constraints to the Maximin Model we get:

maximize y,

subject to

$$A_1 \begin{bmatrix} \mathbf{x} \\ y \end{bmatrix} \geq 0,$$

$$A_2\mathbf{x} = \mathbf{b}_2,$$

$$x_i \in \{0,1\}, \qquad i = 1, \ldots, I,$$

$$y \geq 0,$$

where

$$\begin{bmatrix} \mathbf{x} \\ y \end{bmatrix} = \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_I \\ y \end{bmatrix},$$

$$A_1 = \begin{bmatrix} I_1(\theta_1) & \cdots\cdots\cdots & I_I(\theta_1) & -r_1 \\ \vdots & & & \vdots \\ I_1(\theta_K) & \cdots\cdots\cdots & I_I(\theta_K) & -r_K \end{bmatrix},$$

$$A_2 = \begin{array}{c}
\begin{array}{cccccccccccccccccc} & & 150 & & & & & & 150 & & & & & & 150 & & & \end{array} \\
\begin{bmatrix}
1 & . & . & . & . & 1 & 0 & . & . & . & . & 0 & 0 & . & . & . & . & 0 \\
0 & . & . & . & . & 0 & 1 & . & . & . & . & 1 & 0 & . & . & . & . & 0 \\
0 & . & . & . & . & 0 & 0 & . & . & . & . & 0 & 1 & . & . & . & . & 1 \\
1 & . & 1 & 0 & . & 0 & 1 & . & 1 & 0 & . & 0 & 1 & . & 1 & 0 & . & 0 \\
0 & . & 0 & 1 & . & 1 & 0 & . & 0 & 1 & . & 1 & 0 & . & 0 & 1 & . & 1
\end{bmatrix}, \\
\begin{array}{cccccccccccccccccc} & 80 & & 70 & & & & 80 & & 70 & & & & 80 & & 70 & & \end{array}
\end{array}$$

$$\mathbf{b}_2 = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 15 \\ 15 \end{bmatrix} .$$

We can partition $\underset{\sim}{\pi}^T$ (see Step 1) into $\underset{\sim}{\pi}_1{}^T$ and $\underset{\sim}{\pi}_2{}^T$ where $\underset{\sim}{\pi}_2{}^T$ corresponds to the constraints for the composition of the test. In the same way each column $\mathbf{a}_i$ is partitionable into a part $\mathbf{a}_{1i}$ and $\mathbf{a}_{2i}$. Thus, in Step 2 the reduced costs can be computed by

$$d_i = \underset{\sim}{\pi}_1{}^T \mathbf{a}_{1i} + \underset{\sim}{\pi}_2{}^T \mathbf{a}_{2i} - c_i .$$

For items belonging to the same subdomains we have to compute $\underset{\sim}{\pi}_2{}^T \mathbf{a}_{2i}$ only once. The above implies that CPU time is gained, because less computations are needed. Also, we can save storage capacity, because we have to store $\mathbf{a}_{2i}$ only for groups of items and not for each item separately.

Computational Experience

Two item banks were generated to determine if CPU time is gained due to the modifications. Both item banks contained 450 items. The items of the first item bank fitted the Rasch model ($b_i \sim N(0,1)$) and the items of the other bank fitted

the 3-parameter model ($a_i \sim U(0.5,1.5)$; $b_i \sim N(0,1)$; $c_i = 0.1$). Three kinds of tests were constructed (van der Linden, 1985):

1) Selective tests, that is, tests which give maximal information at a cut-off point at the ability continuum.

2) Classification tests, that is, tests which give maximal information at two or more cut-off points.

3) Diagnostic tests, that is, tests with a flat target information function for a specified interval of the ability continuum.

In the Maximin Model the kind of test is specified by the test constructor by choosing the number of ability levels K, the ability levels θ_k, and the constants r_k. In the numerical experiments these values were specified as follows: (1) $K = 1$; $\theta_1 = 0$; and $r_1 = 1$ (selective test); (2) $K = 2$; $\theta_1 = -1$, $\theta_2 = 1$; and $r_1 = r_2 = 1$ (classification test); and (3) $K = 3$; $\theta_1 = -2$, $\theta_2 = 0$ and $\theta_3 = 2$; and $r_1 = r_2 = r_3 = 1$ (diagnostic test). Observe that the distinction between classification and diagnostic tests is not always clear from the specified values. The constraints in (3) and (4) should be specified explicitly to make the Maximin Model complete. Five different constraint sets were taken into account.

Constraint set 1

This set 1 contains the constraint:

$$(15) \qquad \sum_{i=1}^{450} x_i = 30.$$

Constraint (15) implies that 30 items are selected.

Constraint set 2

This set 2 contains the constraints:

$$(16) \qquad \sum_{i=1}^{450} t_i x_i \leq 1125,$$

along with (10) - (14).

Constraint (16) is a restriction on the administration time. The coefficient t_i ($\sim U(20,60)$) is an estimate of the time needed for answering item i.

Constraint set 3

This set 3 contains the constraints:

$$(17) \qquad \sum_{i=j*10+1}^{j*10+10} x_i \leq 1, \qquad j = 0, \ldots, 44,$$

along with (10) - (15).

Suppose the item bank can be partitioned into subsets of 10 items, such that each item in such a subset contains a cue about the other items in the subset. Constraints (17) prohibit that more than 1 item from such a subset is selected.

Constraint set 4

This set 4 contains the constraints:

$$(18)\qquad \sum_{i=1}^{25} x_i + \sum_{i=51}^{75} x_i + \ldots + \sum_{i=401}^{425} x_i \geq 10,$$

$$(19)\qquad \sum_{i=26}^{50} x_i + \sum_{i=76}^{100} x_i + \ldots + \sum_{i=426}^{450} x_i \leq 20,$$

$$(20)\qquad \sum_{i=1}^{150} x_i \geq 10,$$

$$(21)\qquad \sum_{i=151}^{300} x_i \leq 12,$$

$$(22)\qquad \sum_{i=301}^{450} x_i \leq 8,$$

along with (15) - (16).

Constraints (18) - (22) control the composition of the test.

Constraint set 5

This set 5 contains the constraints:

$$\sum_{i=j*50+1}^{j*50+50} x_i \le 5, \qquad j = 0,\ldots, 8, \tag{23}$$

along with (15), (16), (18) - (22).

The item bank is assumed to be partitioned in subsets of 50 items. According to constraints (23) at most 5 items are selected from these subsets.

The modifications in the revised simplex method were implemented in the computer program LINPROG (Anthonisse, 1984). All experiments were conducted on an Olivetti M24 personal computer with hardcard and without mathematical coprocessor. In the CPU times reported in Tables 1 through 9 the times for reading the input file, for the initialization, and for writing to the output file are not included.

For the P value in Strategy 3 usually an integer value ranging from 2 to 10 is chosen (Lasdon, 1970, p.311). In our experiments the values 5 and 10 were chosen. The P values in Strategy 4 were chosen to be 50, 100, 150, 200, and 250. For higher values of P Strategy 4 can not be much better than Strategy 1, because the computational burden in Step 2 is not much smaller.

In Table 1 the CPU times (in secs.) and numbers of iterations are given for the construction of tests from both item banks. The pricing strategy was varied, and selective, classification and diagnostic tests were constructed.

Insert Table 1 here

Constraint Set 1 was used implying that the modifications for practical constraints were not applied. Strategy 4 with small values of P gave the best results.

In Tables 2 through 9 CPU times (in secs.) and numbers of iterations are given. In each table the item bank and constraint set were fixed while the pricing strategies and the kind of test were varied. In all the tables results for the unmodified as well as the revised simplex method with modification for the practical constraints are given.

Insert Table 2 - 9 here

There should not be a difference in the number of iterations between the modified and unmodified revised simplex method. However, differences did occur and mostly they occured for pricing strategies that allowed for very small increments of

the objective function values. Thus, numerical inprecision may have caused these differences (Obsreve, however, that the optimal solutions eventually were always equal). If numerical problems did not occur, the modified method was always faster than the unmodified method. In all tables the same tendency is seen: Strategy 1 and 2 are in general the worst pricing strategies. The best results gives Strategy 4.

In Table 10 the objective function values and the numbers of variables with fractional values in the optimal solution are given for all generated problems.

Insert Table 10 here

Table 10 shows that most of the practical constraints had some effect on the solution of the problem. Constraints (23), however, were redundant except for one case.

Discussion

The construction of tests by 0-1 linear programming is a new development in item response theory. In this paper some test construction problems based on the Maximin Model were introduced and numerical experiments were conducted. The conclusions in this section are based on the numerical

experiments on the proposed test construction problems with the computer program LINPROG. Although, the Maximin Model was used in the experiments it should be clear that the pricing strategies and modification for practical constraints can also be applied to other test construction models.

The tables show that for an item bank calibrated under the 3-parameter model the tests are constructed faster and that the number of iterations is smaller than for an item bank calibrated under the Rasch model. For the 3-parameter model the differences between the item information functions are larger, which makes it easier for the revised simplex method to make the distinction between "good" and "bad" items.

Only Step 2 in the revised simplex method depends on the pricing strategy. Table 1 through 9 illustrate that it is the most time consuming step and that determining the variable to leave the basis and computing B^{-1} is not so time consuming. The amount of computations to be executed in Step 2 is heavily influenced by the number of variables in the model. For the other steps only the number of constraints is important. In the numerical experiments 450 variables corresponding to the items were present. If this number is increased, the CPU time will probably increase and the larger part of this increment is caused by Step 2. Hence, if the model contains more variables, the gain by using fast pricing strategies for Step 2 will probably be larger.

Strategy 1 does not necessarily need fewer iterations than the other pricing strategies. For instance, in Table 2 the number of iteratiens for Strategy 1 is not the smallest for all kind of tests. This explains the success of Strategy 4: The number of computations per iteration is much smaller especially for low values of P, whereas the number of iterations is in most cases not much larger and sometimes even smaller.

In general Strategy 4 gives the best results, although Strategy 3 is sometimes better (see Table 9). The problem with Strategy 4 is the choice of the P value. Strategies 1 and 2 are special cases of Strategy 4 and do not give good results. From this one can conclude that P should not be too small or too large. The tables show that P = 50 as well as P = 200 and the values in between yield fast CPU times. Of course, the optimal choice of P depends on the number of items in the item bank (P = 200 for a bank with 200 items is not likely to be a good choice).

Beside the pricing strategies mentioned in this paper other strategies are possible (e.g., Goldfarb & Reid, 1977; Harris, 1973; Kuhn & Quandt, 1963). In this paper the choice of pricing strategies was restricted to strategies which are easy to implement in an existing computer program.

The modification for practical constraints is an improvement, except for some cases where the number of iterations between the modified and unmodified method differs. Noc the number of added constraints is important,

but the number of nonzero coefficients in the columns of matrix A, because in LINPROG multiplications with zero are omitted. The improvement in CPU time, for instance, is larger for Constraint Set 5 than for Constraint Set 3 although the number of constraints is larger in the latter. It can be seen that with respect to the pricing strategies the modification is most effective for Strategy 1 and least effective for Strategy 2. The larger the value of P in Strategy 4 the larger the improvement in CPU time.

The number of constraints in (8) is an upper bound for the number of variables with fractional values in the solution of a linear programming problem. From Table 10 a distinction can be made between hard and easy constraints, where the hard constraints play an important role in causing variables with fractional values. In the hard constraints the coefficients are real valued; they correspond with the administration time and the test information function constraints. The easy constraints have coefficients 0 or 1 and an integer as right hand side; they correspond with constraints with respect to the composition of the test. In the combinatorial optimization literature one can find conditions under which the solution of a linear programming problem is guaranteed to be integer (see e.g. Papdimitriou & Steiglitz, 1982; Schrijver, 1986).

References

Adema, J.J., & van der Linden, W.J. (198^\. Algorithms .or computerized test constiuction using classical item parameters. Journal of Education:l Statistics, 14, 279-290.

Adema, J.J., Boekkooi-Timminga, E., & van der Linden, W.J. (in press). Achievement test construction using 0-1 linear programming. European Journal of Operational Research.

Anthonisse, J.M. (1984). LINPROG. Centre for Mathematics and Computer Science: Amsterdam.

Dantzig, G.B. (1963). Linear progr ming and extensions. Princeton, NJ: Princeton University Press.

Goldfarb, D., & Reid, J.K. (1977). A practicable steepest-edge simplex algorithm. Mathematical Programming, 12, 361-371.

Harris, P.M.J. (1973). Pivot selection methods of the DEVEX LP code, Mathematical Programming, 5, 1-28.

Hartley, R. (1985). Linear and nonlinear programming. Chichester: Ellis Horwood Limited.

Kuhn, H.W., & Quandt, R.E. (1963). An experimental study of the simplex method. In N.C. Metropolis et al. (Ed.), Experimental arithmetic, high speed computing and mathematics (Proceedings of symposia in applied mathematics, Vol XV). New York: McGraw Hill.

Land, A.H., & Doig, A.G. (1960). An automated method for solving discrete programming problems. Econometrica, 28, 497-520.

Lasdon, L.S. (1970). Optimization theory for large systems. New York: Macmillan.

Murtagh, B.A. (1981). Advanced linear programming: computation and practice. New York: McGraw Hill.

Papadimitriou, C.H., & Steiglitz, K. (1982). Combinatorial optimization: Algorithms and complexity. Englewood Cliffs, NJ: Prentice-Hall.

Schrijver, A. (1986). Theory of linear and integer programming. Chichester: Wiley.

Syslo, M.M., Deo, N., & Kowalik, J.S. (1983). Discrete optimization algorithms with Pascal programs. Englewood Cliffs, NJ: Prentice-Hall.

Theunissen, T.J.J.M. (1985). Binary programming and test design. Psychometrika, 50, 411-420.

van der Linden, W.J. (1985). Decision theory in educational research and testing. In T. Huesen & T.N. Postlewaite (Eds.), International encyclopedia of education: Research and studies. Oxford: Pergamon Press.

van der Linden, W.J., & Boekkooi-Timminga, E. (1989). A maximin model for test design with practical constraints. Psychometrika, 54, 237-247.

Table 1

Results for the Revised Simplex Method Without Modification under Constraint Set 1

Strategy	p[a]	Rasch: CPU time (secs.)	Rasch: # Iterations	3-parameter: CPU time (secs.)	3-parameter: # Iterations
		Selective Test			
1	n.a.	103	90	103	89
2	n.a.	138	292	134	285
3	5	89	90	89	90
	10	86	90	89	90
4	50	34	118	32	114
	100	39	101	38	101
	150	47	96	49	99
	200	56	93	57	95
	250	68	95	61	88
		Classification Test			
1	n.a.	167	110	168	111
2	n.a.	423	767	166	250
3	5	150	139	101	118
	10	190	177	118	130
4	50	136	213	62	137
	100	110	165	79	135
	150	114	148	86	121
	200	148	157	103	122
	250	125	125	114	117
		Diagnostic Test			
1	n.a.	214	120	163	97
2	n.a.	466	384	380	347
3	5	191	158	140	120
	10	182	163	108	115
4	50	95	159	82	144
	100	105	146	78	117
	150	123	140	110	129
	200	136	133	125	126
	250	151	129	127	116

[a] n.a. = not applicable

Table 2

Results for the Revised Simplex Method under Constraint Set 2 for the Item Bank Calibrated under the Rasch model

		Unmodified		Modified	
Strategy	P[a]	CPU time (secs.)	# Iterations	CPU time (secs.)	# Iterations
		Selective Test			
1	n.a.	298	153	242	153
2	n.a.	430	501	407	501
3	5	233	174	199	174
	10	247	195	212	195
4	50	152	192	141	192
	100	93	193	125	163
	150	153	155	139	158
	200	164	147	138	147
	250	203	156	171	156
		Classification Test			
1	n.a.	433	177	368	177
2	n.a.	587	807	561	833
3	5	430	230	385	230
	10	514	275	470	275
4	50	290	240	403	289
	100	220	195	334	248
	150	232	176	209	176
	200	291	187	261	187
	250	333	183	277	183
		Diagnostic Test			
1	n.a.	485	176	420	176
2	n.a.	395	611	394	675
3	5	413	217	374	217
	10	393	222	358	222
4	50	341	242	328	242
	100	239	186	222	186
	150	348	207	322	207
	200	223	148	198	148
	250	304	165	270	165

[a] n.a. = not applicable

Table 3

Results for the Revised Simplex Method under Constraint Set 2 for the Item Bank Calibrated under the 3-parameter model

Strategy	P[a]	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	225	127	179	127
2	n.a.	427	434	396	458
3	5	178	135	145	135
	10	164	150	137	150
4	50	95	150	86	150
	100	82	164	111	149
	150	120	132	101	130
	200	140	129	117	129
	250	169	137	141	137
		Classification Test			
1	n.a.	295	135	245	135
2	n.a.	533	528	513	533
3	5	333	199	296	199
	10	215	164	186	164
4	50	144	168	222	214
	100	238	199	192	184
	150	242	181	217	181
	200	273	177	243	177
	250	248	158	216	158
		Diagnostic Test			
1	n.a.	384	150	329	150
2	n.a.	410	695	401	723
3	5	361	195	324	195
	10	323	199	291	199
4	50	216	190	206	190
	100	212	171	197	171
	150	281	183	257	183
	200	208	142	185	142
	250	236	142	207	142

[a] n.a. = not applicable

Table 4

Results for the Revised Simplex Method under Constraint Set 3 for the Item Bank Calibrated under the Rasch model

Strategy	P[a]	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	358	147	288	147
2	n.a.	339	302	329	302
3	5	213	151	170	151
	10	206	178	167	178
4	50	138	151	125	151
	100	196	169	173	167
	150	157	128	127	147
	200	146	119	122	119
	250	217	159	155	128
		Classification Test			
1	n.a.	571	198	479	198
2	n.a.	574	655	559	655
3	5	338	231	284	231
	10	339	241	289	241
4	50	212	174	200	174
	100	324	207	329	237
	150	279	168	251	168
	200	296	200	257	200
	250	320	200	363	200
		Diagnostic Test			
1	n.a.	511	180	428	180
2	n.a.	422	527	411	527
3	5	324	190	276	190
	10	288	195	247	195
4	50	285	255	267	255
	100	245	189	225	192
	150	288	194	257	194
	200	312	184	275	184
	250	303	163	294	179

[a] n a. = not applicable

Table 5

Results for the Revised Simplex Method under Constraint Set 3 for the Item Bank Calibrated under the 3-parameter model

Strategy	p^a	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	217	104	169	104
2	n.a.	294	278	283	278
3	5	174	124	137	124
	10	138	113	108	113
4	50	93	117	83	117
	100	111	121	97	121
	150	134	118	121	136
	200	117	101	95	101
	250	159	124	142	116
		Classification Test			
1	n.a.	312	123	253	123
2	n.a.	396	384	386	384
3	5	227	156	189	156
	10	245	178	205	178
4	50	306	220	288	220
	100	310	195	281	193
	150	267	167	241	167
	200	225	157	194	157
	250	240	153	260	152
		Diagnostic Test			
1	n.a.	457	160	382	160
2	n.a.	468	583	456	583
3	5	282	155	238	155
	10	294	188	253	188
4	50	241	214	226	214
	100	196	155	179	155
	150	206	134	183	134
	200	284	165	250	165
	250	340	178	235	148

[a] n.a. = not applicable

Table 6

Results for the Revised Simplex Method under Constraint Set 4 for the Item Bank Calibrated under the Rasch model

		Unmodified		Modified	
Strategy	p[a]	CPU time (secs.)	# Iterations	CPU time (secs.)	# Iterations
		Selective Test			
1	n.a.	194	110	151	110
2	n.a.	338	320	328	320
3	5	177	138	146	138
	10	171	148	145	148
4	50	110	153	100	153
	100	130	148	115	148
	150	136	144	115	144
	200	139	126	114	126
	250	151	126	123	126
		Classification Test			
1	n.a.	340	144	284	144
2	n.a.	433	655	421	655
3	5	383	206	339	206
	10	328	212	299	212
4	50	342	257	290	240
	100	309	221	287	221
	150	325	209	294	209
	200	374	213	333	213
	250	291	172	254	172
		Diagnostic Test			
1	n.a.	350	139	295	139
2	n.a.	247	414	238	414
3	5	302	179	269	179
	10	278	185	252	185
4	50	192	195	181	195
	100	204	165	187	165
	150	247	168	223	168
	200	302	174	269	174
	250	275	152	240	152

[a] n.a. = not applicable

Table 7

Results for the Revised Simplex Method under Constraint Set 4 for the Item Bank Calibrated under the 3-parameter model

Strategy	P[a]	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	179	104	140	104
2	n.a.	456	364	445	364
3	5	140	124	116	124
	10	120	123	102	123
4	50	97	145	88	145
	100	95	125	82	125
	150	123	131	102	131
	200	103	106	83	106
	250	121	106	96	106
		Classification Test			
1	n.a.	274	124	226	124
2	n.a.	388	437	379	437
3	5	273	184	244	184
	10	163	145	145	145
4	50	182	186	186	194
	100	134	140	119	140
	150	155	135	134	135
	200	175	136	150	136
	250	237	148	204	148
		Diagnostic Test			
1	n.a.	240	104	199	104
2	n.a.	230	395	222	395
3	5	222	140	194	140
	10	207	147	185	147
4	50	163	163	154	163
	100	191	160	175	160
	150	187	144	167	144
	200	18[illegible]	131	164	131
	250	246	144	213	144

[a] n.a. = not applicable

Table 8

Results for the Revised Simplex Method under Constraint Set 5 for the Item Bank Calibrated under the Rasch model

Strategy	p^a	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	263	129	193	129
2	n.a.	381	463	363	516
3	5	140	155	116	167
	10	138	172	104	172
4	50	118	148	107	148
	100	159	154	140	154
	150	188	163	157	163
	200	198	152	161	152
	250	186	133	148	133
		Classification Test			
1	n.a.	472	178	377	178
2	n.a.	398	1005	348	914
3	5	246	223	195	227
	10	207	222	164	222
4	50	204	249	186	249
	100	222	215	197	215
	150	254	184	219	184
	200	273	176	234	176
	250	330	199	275	199
		Diagnostic Test			
1	n.a.	381	151	301	151
2	n.a.	257	633	209	573
3	5	253	213	201	210
	10	228	233	208	233
4	50	135	192	121	192
	100	170	177	146	177
	150	200	176	168	176
	200	199	142	166	142
	250	282	175	231	175

[a] n.a. = not applicable

Table 9

Results for the Revised Simplex Method under Constraint Set 5 for the Item Bank Calibrated under the 3-parameter model

Strategy	P[a]	Unmodified CPU time (secs.)	Unmodified # Iterations	Modified CPU time (secs.)	Modified # Iterations
		Selective Test			
1	n.a.	218	111	159	111
2	n.a.	390	524	360	519
3	5	98	116	85	137
	10	85	119	64	119
4	50	132	159	120	159
	100	150	150	129	150
	150	139	130	114	130
	200	120	110	94	110
	250	193	139	155	139
		Classification test			
1	n.a.	349	142	274	142
2	n.a.	253	522	250	557
3	5	190	180	128	161
	10	142	168	111	168
4	50	175	199	160	199
	100	198	179	176	179
	150	255	177	221	177
	200	207	136	176	136
	250	217	132	181	132
		Diagnostic Test			
1	n.a.	335	133	263	133
2	n.a.	245	614	219	581
3	5	240	202	164	177
	10	209	199	171	199
4	50	117	163	105	163
	100	165	169	142	169
	150	176	153	148	153
	200	205	143	170	143
	250	214	137	176	137

[a] n.a. = not applicable

Table 10

Objective Function Values and Number of Fractional Values in the Solutions of the Problems Corresponding to Table 1 through 9

	Selective		Classification		Diagnostic	
Table[a]	Obj. Func. Value	# Frac. Values	Obj. Func. Value	# Frac. Values	Obj. Func. Value	# Frac Values
1 (R)	7.4948	0	5.[illegible]969	2	4.0073	3
1 (3p)	12.2783	0	7.5496	2	4.7022	3
2	7.4915	4	5.8960	6	4.0048	6
3	12.2633	4	7.5344	6	4.6805	5
4	7.4864	2	5.8945	4	3.9928	8
5	11.9881	2	7.4521	4	4.5462	6
6	7.4916	2	5.8959	3	4.0049	4
7	12.2726	2	7.5496	2	4.6648	5
8	7.4916	2	5.8959	3	4.0048	6
9	12.2726	2	7.5496	2	4.6606	5

[a] The table r bers are used to identify the problem at hand. In Table 1 results for an item bank calibrated under the Rasch model (R) and a bank calibrated under the 3-parameter model (3p) are given.

Titles of recent Research Reports from the Division of Educational Measurement and Data Analysis, University of Twente, Enschede, The Netherlands.

RR-90-5	J.J. Adema, *A Revised Simplex Method for Test Construction Problems*
RR-90-4	J.J. Adema, *Methods and Models for the Construction of Weakly Parallel Tests*
RR-90-3	H.J. Vos, *Simultaneous Optimization of Classification Decisions Followed by an End-of-Treatment Test*
RR-90-2	H. Tobi, *Item Response Theory at subject- and group-level*
RR-90-1	P. Westers & H. Kelderman, *Differential item functioning in multiple choice items*
RR-89-6	J.J. Adema, *Implementations of the Branch-and-Bound method for test construction problems*
RR-89-5	H.J. Vos, *A simultaneous approach to optimizing treatment assignments with mastery scores*
RR-89-4	M.P.F. Berger, *On the efficiency of IRT models when applied to different sampling designs*
RR-89-3	D.L. Knol, *Stepwise item selection procedures for Rasch scales using quasi-loglinear models*
RR-89-2	E. Boekkooi-Timminga, *The construction of parallel tests from IRT-based item banks*
RR-89-1	R.J.H. Engelen & R.J. Jannarone, *A connection between item/subtest regression and the Rasch model*
RR-88-18	H.J. Vos, *Applications of decision theory to computer based adaptive instructional systems*
RR-88-17	H. Kelderman, *Loglinear multidimensional IRT models for polytomously scored items*
RR-88-16	H. Kelderman, *An IRT model for item responses that are subject to omission and/or intrusion errors*
RR-88-15	H.J. Vos, *Simultaneous optimization of decisions using a linear utility function*
RR-88-14	J.J. Adema, *The construction of two-stage tests*
RR-88-13	J. Kogut, *Asymptotic distribution of an IRT person fit index*

RR-88-12 E. van der Burg & G. Dijksterhuis, *Nonlinear canonical correlation analysis of multiway data*

RR-88-11 D.L. Knol & M.P.F. Berger, *Empirical comparison between factor analysis and item response models*

RR-88-10 H. Kelderman & G. Macready, *Loglinear-latent-class models for detecting item bias*

RR-88-9 W.J. van der Linden & T.J.H.M. Eggen, *The Rasch model as a model for paired comparisons with an individual tie parameter*

RR-88-8 R.J.H. Engelen, W.J. van der Linden, & S.J. Oosterloo, *Item information in the Rasch model*

RR-88-7 J.H.A.N. Rikers, *Towards an authoring system for item construction*

RR-88-6 H.J. Vos, *The use of decision theory in the Minnesota Adaptive Instructional System*

RR-88-5 W.J. van der Linden, *Optimizing incomplete sample designs for item response model parameters*

RR-88-4 J.J. Adema, *A note on solving large-scale zero-one programming problems*

RR-88-3 E. Boekkooi-Timminga, *A cluster-based method for test construction*

RR-88-2 W.J. van der Linden & J.J. Adema, *Algorithmic test design using classical item parameters*

RR-88-1 E. van der Burg & J. de Leeuw, *Nonlinear redundancy analysis*

Research Reports can be obtained at costs from Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.

...tment of
EDUCATION

A publication by
the Department of Education
of the University of Twente
de
nds